Batteries in Space

Designing Energy-Optimal Satellites with Statistical Model Checking

Wognsen, Erik Ramsgaard

DOI (link to publication from Publisher):
10.5278/vbn.phd.engsci.00072

Publication date:
2016

Document Version
Publisher's PDF, also known as Version of record

BATTERIES IN SPACE: DESIGNING ENERGY-OPTIMAL SATELLITES WITH STATISTICAL MODEL CHECKING

BY
ERIK RAMSGAARD WOGNSEN

DISSERTATION SUBMITTED 2016

AALBORG UNIVERSITY
DENMARK

Batteries in Space: Designing Energy-Optimal Satellites with Statistical Model Checking

Ph.D. Dissertation

Erik Ramsgaard Wognsen

Dissertation submitted December, 2015

Thesis submitted:	December, 2015
PhD supervisor:	Prof. Kim Guldstrand Larsen Aalborg University
Assistant PhD supervisor:	Assoc. Prof. René Rydhof Hansen Aalborg University
PhD committee:	Assoc. Prof. Bent Thomsen (chairman) Aalborg University
	Prof. Paul Petterson Mälardalen University
	Prof. Jan Madsen DTU Informatics
PhD Series:	Faculty of Engineering and Science, Aalborg University

ISSN (online): 2246-1248
ISBN (online): 978-87-7112-451-4

Published by:
Aalborg University Press
Skjernvej 4A, 2nd floor
DK – 9220 Aalborg Ø
Phone: +45 99407140
aauf@forlag.aau.dk
forlag.aau.dk

Printed in Denmark by Rosendahls, 2016

Abstract

Satellites provide humanity with many useful services, but for the services to work reliably, satellites must be carefully designed, programmed and verified. Formal methods provide many techniques to analyze, check, prove, and synthesize systems. But in addition to correctness we are also interested in energy and how it is used. This thesis follows the transition of formal methods research into quantitative territory.

Energy is scarce in space, so to get the most out of the satellites we have spent large sums to place in orbit, we must also understand how they use energy. We treat three topics, starting with batteries.

We show that precise modeling of battery behavior in the context of formal methods enables more efficient operation without extensive safety margins, and that battery-aware scheduling can reduce energy waste. On the other hand, pushing a battery to the limit to provide optimal short term performance reduces long term battery life over hundreds of recharge cycles. We provide a method to evaluate the long term effect of proposed usage profiles on the battery and weigh them against the benefit of the increased short term performance.

The second topic concerns the actual energy use in satellite equipment such as radio transceivers. Many electronic circuits and computations can be expressed as dataflow graphs. First we show how a translation of dataflow graphs to priced timed automata enables the use of cost-optimal reachability algorithms to perform energy-optimal scheduling, and how this can be used to study trade-offs between time and energy. For the specific dataflow formalism finite-state machine-based scenario-aware dataflow, we develop a systematic translation to timed automata such that general properties of the dataflow graphs can be model checked.

The final topic is the correct operation of computer processors in space. We formalize a realistic low-level assembly language and show how programs in it can be modified to guarantee detection of computation errors caused by transient bit errors in data registers, thus making satellite software more resilient to high-energy particles found outside our atmosphere as well as aggressive power saving techniques.

Resumé

Satellitter forsyner os med mange nyttige services, men for at disse kan fungere pålideligt, må satellitterne være grundigt designet, programmeret og verificeret. Formelle metoder giver mange teknikker til at analysere, tjekke, bevise og syntetisere systemer. Men ud over korrekt opførsel er vi også interesseret i energy, og hvordan det bliver brugt. Denne afhandling følger overgangen for forskningen i formelle metoder til det kvantitative domæne.

Energien er knap i rummet, så for at få mest muligt ud af de satellitter, vi har brugt store summer på at opsende, må vi også forstå, hvordan de bruger energi. Vi behandler tre emner og lægger ud med batterier.

Vi viser at præcis modellering af batteriopførsel i kontekst af formelle metoder tillader mere effektiv drift uden omfattende sikkerhedsmarginer, samt at planlægning med opmærksomhed på batterier kan reducere energispild. På den anden side kan det at presse et batteri til det yderste for at opnå optimal ydelse på kort sigt reducere batteriets levetid set over hundredevis af opladningscyklusser. Vi beskriver en metode til at evaluere de langsigtede virkninger potentielle brugsprofiler til have på batteriet, og til at veje dem op mod fordelene ved forøget ydelse på kort sigt.

Det andet emne handler om egentligt energiforbrug i satellitudstyr så som radiosendere og -modtagere. Mange elektroniske kredsløb og beregninger kan udtrykkes som dataflow-grafer. Først viser vi, hvordan en oversættelse af dataflow-grafer til prismærkede tidsautomater tillader anvendelsen af algoritmer til omkostningsoptimal reachability til at udføre energi-optimal planlægning. Dette kan også bruges til at afprøve afvejninger mellem tids- og energiforbrug. For dataflow-formalismen "finite-state machine-based scenario-aware dataflow" udvikler vi en systematisk oversættelse til tidsautomater, således at generelle egenskaber ved dataflow-graferne kan modeltjekkes.

Det sidste emne er korrekt opførsel af computerprocessorer i rummet. Vi formaliserer et realistisk lavniveau-sprog og viser, hvordan programmer i dette kan modificeres for at garantere detektion af beregningsfejl forårsaget af kortvarige bitfejl i dataregistre. Denne teknik kan gøre satellitprogrammel mere modstandsdygtigt over for højenergipartikler, der findes uden for vores atmosfære, såvel som over for aggressive strømsparingsteknikker.

Contents

Thesis Details

Thesis Title:	Batteries in Space: Designing Energy-Optimal Satellites with Statistical Model Checking
Ph.D. Student:	Erik Ramsgaard Wognsen
Supervisors:	Prof. Kim Guldstrand Larsen, Aalborg University Assoc. Prof. René Rydhof Hansen, Aalborg University

The main body of this thesis consist of the following papers.

[A] Erik Ramsgaard Wognsen, René Rydhof Hansen, Kim Guldstrand Larsen, "Battery-Aware Scheduling of Mixed Criticality Systems," *Leveraging Applications of Formal Methods, Verification and Validation (ISoLA 2014)* Specialized Techniques and Applications, Part II., LNCS vol. 8803, pp. 208–222, 2014.

[B] Erik Ramsgaard Wognsen, Boudewijn R. Haverkort, Marijn Jongerden, René Rydhof Hansen, Kim Guldstrand Larsen, "A Score Function for Optimizing the Cycle-Life of Battery-Powered Embedded Systems," *Formal Modeling and Analysis of Timed Systems (FORMATS 2015)*, LNCS vol. 9268, pp. 305–320, 2015.

[C] Erik Ramsgaard Wognsen, René Rydhof Hansen, Kim Guldstrand Larsen, Peter Koch "Energy-Aware Scheduling of FIR Filter Structures," *To be submitted*.

[D] Mladen Skelin, Erik Ramsgaard Wognsen, Mads Chr. Olesen, René Rydhof Hansen, Kim Guldstrand Larsen, "Model Checking of Finite-state Machine-based Scenario-aware Dataflow Using Timed Automata," *Industrial Embedded Systems (SIES 2015)*, pp. 235–244, 2015.

[E] René Rydhof Hansen, Kim Guldstrand Larsen, Mads Chr. Olesen, Erik Ramsgaard Wognsen, "Formal Methods for Modelling and Analysis of Single-Event Upsets," To appear in *Information Reuse and Integration (IRI 2015) workshop papers*.

In addition to the main papers, the following publications have also been made.

- Erik Ramsgaard Wognsen, Henrik Søndberg Karlsen, Marcus Calverley, Mikkel Normann Follin, Bent Thomsen, Hans Hüttel, "A Secure Relay Protocol for Door Access Control," *Brazilian Symposium on Information and Computer System Security (SBSeg 2012)*, pp. 196–209, 2012.

- Erik Ramsgaard Wognsen, Henrik Søndberg Karlsen, Mads Chr. Olesen, René Rydhof Hansen, "Formalisation and Analysis of Dalvik Bytecode," *Science of Computer Programming*, vol. 92, pp. 25–55, 2014.

- Mladen Skelin, Erik Ramsgaard Wognsen, Mads Chr. Olesen, René Rydhof Hansen, Kim Guldstrand Larsen, "Towards Translating FSM-SADF to Timed Automata," *Investigating Dataflow in Embedded computing Architectures (IDEA 2015)*, Computer Science Reports 1502, Technische Universiteit Eindhoven, pp. 13–16, 2015.

- Erik Ramsgaard Wognsen, Marijn Jongerden, Boudewijn R. Haverkort, "A Score Function for State-of-Charge Profiles for Rechargeable Batteries," DESIGN&ELEKTRONIK-Entwicklerforum Batterien & Ladekonzepte, informal proceedings, 2015.

This thesis has been submitted for assessment in partial fulfillment of the PhD degree. The thesis is based on the submitted or published scientific papers which are listed above. Parts of the papers are used directly or indirectly in the extended summary of the thesis. As part of the assessment, co-author statements have been made available to the assessment committee and are also available at the Faculty. The thesis is not in its present form acceptable for open publication but only in limited and closed circulation as copyright may not be ensured.

Preface

This thesis is the result of my three years as a PhD student at Aalborg University. Several people have contributed to its existence: First, I would like to thank my supervisors, Kim Guldstrand Larsen and René Rydhof Hansen, for their guidance during my work, and for making it possible.

The work in this thesis would also have been impossible without my (other) co-authors Mads Chr. Olesen, Mladen Skelin, Peter Koch, Marijn Jongerden, and Boudewijn R. Haverkort. I would also like to thank Boudewijn and Marijn for being my excellent hosts during my three months at Twente University in the Netherlands. Also thanks to the rest of DACS at Twente, and to Philip Hölzenspies for the enjoyable conversations.

Back at Aalborg University I have enjoyed Simon Laursen's company as we have shared the ups and downs of life as PhD students. We have also shared our lunches with the rest of the "lunch club" and our bottles of rum with the "rum club". Thank you for the pleasant moments over the years to Mikael, Line, Henrik, Louise, Peter, Jakob, and Thibaut. My work has been done as part of the SENSATION project. Here, I have enjoyed the meetings and discussions with Holger Hermanns, Gilles Nies, and Jan Krčál of Saarland University, and Peter Bak, David Gerhardt, and Morten Bisgaard of GomSpace ApS.

Finally, I met Aistė around the same time I started my PhD. It has been wonderful sharing the last three years with you, and I hope we will share many, many more.

Erik Ramsgaard Wognsen
Aalborg University, December 31, 2015

Part I

Introduction

Introduction

Since the launch of Sputnik 1, the first human-made satellite, 57 years ago, more than 6000 satellites have been placed in orbit to provide society with services such as communication, navigation, and observation of Earth and space.

We rely on satellite services to function continuously. The malfunction of communication services can be highly problematic, and the absence of weather prediction and military air photos can be directly dangerous. In addition, satellites are expensive to deploy and nearly impossible to access for maintenance. Therefore, the correct functioning and reliability of satellites is an important goal.

To meet this goal, *formal methods* are required: Mathematical techniques to model and verify software and hardware systems. The research community has extensive expertise on analyzing and verifying embedded systems using formal methods. The question treated in this thesis is:

> How can we expand our existing tools and techniques for analyzing embedded systems to analyze satellites under their challenges of limited energy supply and high risk of computation errors?

Satellites harvest energy from the sun, so the size of the solar panels constrain the energy supply. Surplus energy is stored in batteries, which are electrochemical devices that convert energy between electrical and chemical forms. The rate at which this conversion can happen is limited and depends on the use of the battery in the past. Thus, the analysis of satellites depends on the analysis of batteries.

Furthermore, satellites operate without the protection of the atmosphere, which means that radiation and high-energy particles can upset the computations in processors. Computations also risk calculation errors when saving energy by reducing the supply voltage. Therefore another goal is to examine how formal methods can be employed to improve the resilience of embedded systems toward these computation errors.

When the analysis methods have been expanded to more naturally work with battery-powered satellites, the next step is to enable system developers

to use those methods. An important part of satellite communication and radio circuitry is digital signal processing (DSP). DSP can be modelled using *dataflow formalisms*. Therefore we have the final goal of enabling dataflow formalisms to be used together with the methods that are relevant for analyzing battery-powered systems.

Organization The rest of this thesis is organized as an extended introduction followed by a collection of papers. Section 1 presents battery modeling, Section 2 presents model checking, Section 3 presents timed automata formalisms, while Section 4 concludes and gives future perspectives.

Paper A presents a hybrid timed automaton battery model and combines it with a task set. The paper demonstrates how a battery-aware task scheduler can reduce energy waste in a satellite.

Paper B develops a function to evaluate the impact of the short term battery use on the long term wear to the battery. We combine this function with a model of the GomSpace GOMX-3 nanosatellite and show how reinforcement learning can be used to perform a trade-off between short and long term battery use. GomSpace is a Danish company that supplies satellite products and services. Their challenges have in general inspired the work in this thesis and in particular this paper.

Paper C performs energy-aware scheduling of filter computations for satellites' software defined radio (SDR). We translate the filters' dataflow graphs to precedence graphs, and from there to priced timed automata. We use cost-optimal reachability to find schedules and demonstrate a trade-off between time and energy usage.

Paper D describes a general translation of any synchronous dataflow to timed automata to enable systems modeled as dataflow to be used with the timed automata based battery and battery wear models from papers A and B.

Paper E defines and formalizes a low-level language that captures the essential features of the ARM assembly language. We augment it with a special, atomic blue/green branch instruction and then define a static program analysis to prove that a specifically designed program in our low-level language guarantees that a single-event upset ("bitflip") in a data register cannot lead to unnoticed incorrect operation. We then develop a program fragment that can replace the special instruction using only the typical core instructions found on real-world hardware. We employ exhaustive model checking to prove its correctness, and finally, use statistical model checking to quantify the effects of more severe types of single-event upsets.

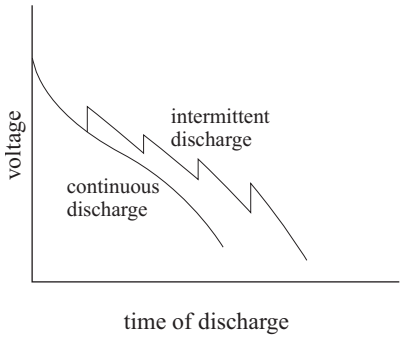

Fig. 1: The recovery effect [20]. With intermittent discharges, the battery can recover while idle.

1 Battery Modeling

Satellites use energy for computation and for interaction with the world through sensors and actuators. This energy is typically delivered by batteries and solar panels[1]. When the solar panels generate more power than is consumed, the surplus is used to charge the batteries, and when the solar panels generate less energy than needed, the battery supplies the difference. However, to understand exactly when happens when a battery is charged and discharged, we must consider its inner workings.

Batteries consist of one or more *electrochemical cells* in which chemically stored energy is released as electrical energy through an electrochemical reaction. Several factors influence the speed of such a reaction and in turn the behavior of the battery. With a battery model, this behavior can be predicted.

One example of the dynamics of batteries from everyday life is the flashlight that is becoming dim as the battery is drained. However, after switching off the dim flashlight for a while, its light will shine stronger when it is again switched on. This is the so-called *recovery effect*, illustrated in Figure 1. Another prevalent effect is the *rate-capacity effect*, see Figure 2, which states that the total amount of energy you can extract from a battery decreases when you increase the rate of extraction.

1.1 Types of Battery Models

Jongerden and Haverkort [19] study various types of battery models including electrochemical models, electrical-circuit models, and analytical models.

[1] It is possible for spacecraft to be nuclear powered (see https://en.wikipedia.org/wiki/Radioisotope_thermoelectric_generator), but this is problematic and generally not done for satellites in Earth orbit because they could spread radioactive material when they eventually deorbit and burn up in the atmosphere.

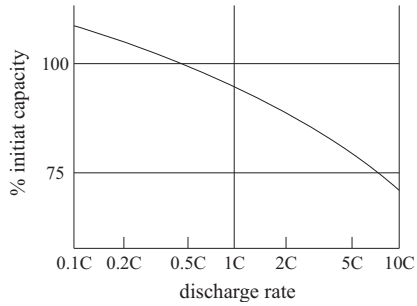

Fig. 2: The rate-capacity effect [20]. Effective capacity is given relative to the ideal capacity at the 2 hour discharge rate, 0.5C. At the $\frac{1}{8}$ hour rate (8C), only 75% of the battery's energy can be extracted before the battery dies.

Electrochemical models describe the actual chemical processes that occur in the battery. As such, they are very precise, but they also depend on many parameters describing the chemical properties and physical dimensions of the components of the electrochemical cells. Such models are developed for several battery technologies including alkaline [24], nickel-cadmium (NiCd) [13], and lithium-ion [14–16]. The program "dualfoil"[2] implements models for simulating lithium-ion, sodium-ion, and nickel-metal hydride batteries. Dualfoil requires choosing over 50 parameters, but due to its high precision, it is sometimes used as comparison to test the precision of other models, because it is much faster, and arguably safer, than testing with real batteries.

Electrical-circuit models imitate the behavior of batteries using (models of) electrical circuits composed of voltage sources, resistors, capacitors, etc. Models have been made for nickel-cadmium, lead-acid and alkaline [18], as well as lithium-ion batteries [17]. These models are simpler than the electrochemical models but still require a considerable effort to configure [19], and may have errors up to 12% in predicting battery lifetime [17].

Whereas electrochemical and electrical-circuit models describe the inner workings of batteries, analytical models describe their effects at a higher level of abstraction. This allows for much more computationally viable simulations and solutions to equations, while still describing battery behavior with good precision. Analytical models range from the simple Peukert's law to Rakhmatov and Vrudhula's diffusion model [25]. In between these two models, Jongerden and Haverkort [19] find the *kinetic battery model* [23] to be well suited for combination with a workload model because it is fairly simple, yet accurate.

[2] http://www.cchem.berkeley.edu/jsngrp/fortran.html

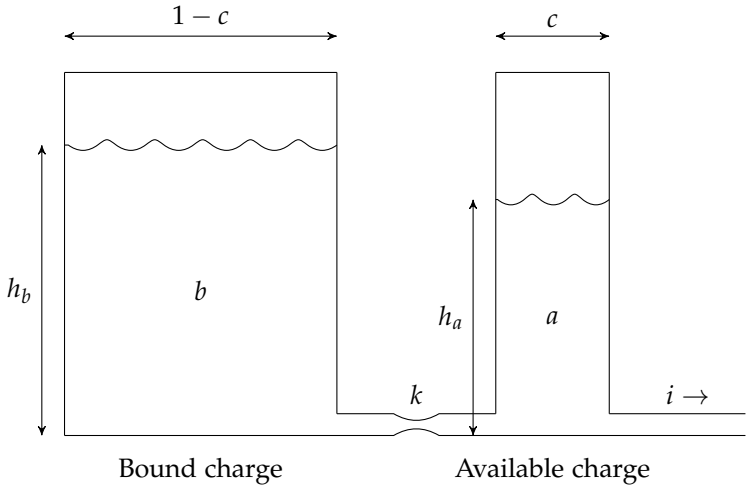

Fig. 3: Illustration of the kinetic battery model

1.2 The Kinetic Battery Model

The kinetic battery model makes a distinction between the charge that is immediately available to drive a given load, and the charge that, over time, can become available. These two portions of the charge are known as the *available* and the *bound* charge, respectively. They are illustrated in Figure 3. In this illustration, the water represents electric charge, and gravity works to balance out the energy stored in the two wells. However, the wells do not represent the physical layout of an electrochemical cell; it is merely a mental model (and batteries also work upside down).

In this model, charge leaves the battery at the rate i. As this happens, the water level h_a drops. This in turn causes bound charge to flow through the valve to become available, such that h_a and h_b approach each other.

The amount of charge represented by h_a and h_b depend on the parameter $0 < c < 1$ such that $a = ch_a$ and $b = (1-c)h_b$. The rate of flow from bound to available charge $k(h_b - h_a)$ is proportional to the height difference, with proportionality factor being the constant k. In sum, the charges are described by this system of differential equations:

$$\begin{cases} \dfrac{\mathrm{d}a}{\mathrm{d}t} = -i + k(h_b - h_a) = -i + k(\dfrac{b}{1-c} - \dfrac{a}{c}) \\ \dfrac{\mathrm{d}b}{\mathrm{d}t} = -k(h_b - h_a) = -k(\dfrac{b}{1-c} - \dfrac{a}{c}) \end{cases} \tag{1}$$

The parameters c and k are constants that depend on the battery technology and can be fitted experimentally. The remaining quantities change over

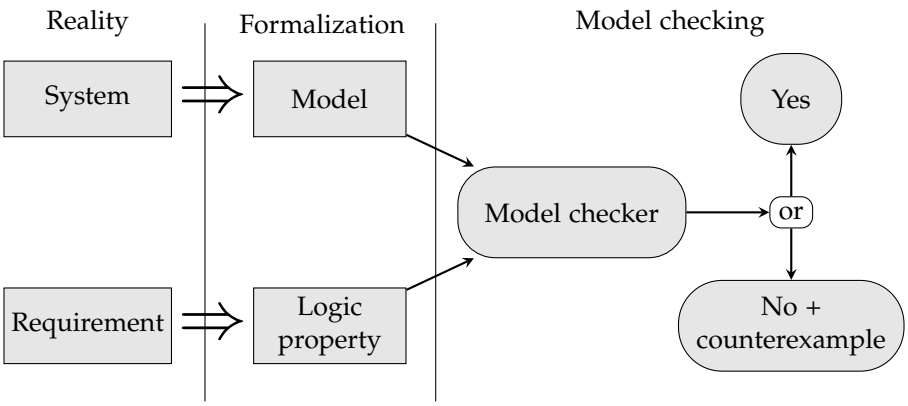

Fig. 4: A model checker takes as input (1) a formal model based on a real system, and (2) a requirement formalized as a logic property. As output it produces either the answer 'Yes', or a 'No' and a counterexample (such as a trace) demonstrating how to violate the desired property.

time, as described by (1). The initial condition is the equilibrium, $h_a = h_b$, where the total capacity C of the battery is distibuted into the available charge cC and the bound charge $(1-c)C$. When $a = h_a = 0$, the battery is considered empty because it cannot readily supply charge, even though it does contain bound charge.

This model, which we will see in action in Paper A, is a good approximation of real battery behavior while being reasonably simple. Specifically, unlike more complicated models such as Dualfoil, the kinetic battery model can be used with the model checking approach.

2 Model Checking

The purpose of verification is to prove the correctness of systems, or to prove that *behavior* satisfies *specification*. The purpose of model checking is to verify that a formal model of a system satisfies a property by exploring the complete state space of the model. This has three components: The model, the properties, and the model checking process itself. See Figure 4.

A model is specified in a language or *formalism*. These include process algebra, state machines/automata, and Petri nets. In this thesis, the focus is on variants of automata that include notions of time. They are the topic of Section 3.

The property or properties can be expressed in different ways. If the property is expressed in the same language as the system model, verification entails checking the *equivalence* of the two models [1, 4]. This method, which predates model checking, has the drawback that the whole behavior of the

system must be embodied in a single model, which could be as complicated to design as the system model itself. Model checking, on the other hand, checks the system model against individual properties, each of which only concern a single aspect of behavior. Properties are expressed in logics such as Hennessy-Milner logic, computational tree logic (CTL), and linear temporal logic (LTL). These are temporal logics that can express properties involving time. In daily language, this would be statements involving words such as 'always', 'eventually', and 'until'.

With model and property in place, model checking can be performed: The statespace of the model is explored to decide whether the model satisfies the property or not. Model checking often employs abstraction techniques. If the statespace is infinite, abstraction is necessary to turn the statespace into a finite, enumerable statespace. But often, it is also desirable to abstract any statespace into a smaller one. This is because statespaces are often unmanageably large due to combinatorial explosion. The statespace consists of the combinations of each state every part of the model can be in. If a system consists of n components that can each be in one of m states, the combined will have m^n states. And that means up to m^n states will have to be explored to formally guarantee that the property holds. Thus, model checking is computationally very expensive for large models. This is known as the *statespace explosion problem*.

In the face of the statespace explosion problem, one might turn to *statistical model checking* [11, 22]. SMC trades absolute guarantees for speed and scalability. Instead of exhaustive statespace exploration, the behavior of the formal model is *simulated* repeatedly. Statistical methods are used to study the resulting simulation *runs*, and produce answers in the form of probabilities rather than Boolean values. For simulation to be possible, the model must be given a stochastic semantics. It is possible to give a non-deterministic model stochastic semantics, as well as to statistically model check an inherently stochastic model. We will see more about this on a concrete formalism in Section 3.2.

Instead of using model checking to determine whether a (laboriously created) model exhibits a specific desired behavior, it can be attractive to algorithmically derive a model that displays the same good behavior. This falls under the field of game theory in computer science. The protagonist aims to find a trajectory that satisfies the desired property while the antagonist tries to prevent the protagonist from succeeding. The (meta) goal for games is to *synthesize* a winning strategy for the protagonist, if possible [5]. This strategy corresponds to a winning subset of behavior in the original, coarser model of all possible behavior.

As with model checking, synthesis is also relevant for stochastic models [21]. Here, simulation can also be used as an alternative or complement to the computationally expensive exhaustive synthesis. By combining simu-

lation with reinforcement learning, it is possible to gradually improve a strategy such that the probability of satisfying a given condition is maximized or minimized [9, 10]. This is used extensively in Paper B.

3 Timed Automata

A central family of formalisms used in model checking, statistical model checking, and games is *timed automata* [2] and variations hereof. A timed automaton is a finite state machine extended with real-valued *clocks*. Transitions can be allowed or disallowed depending on the values of these clocks, and transitions can also reset their value. Time transitions let the clock values grow while remaining in the same discrete state of the system. The (full) state of the system consists of the discrete state, called *location*, and the values of the clocks. First we give a formal definition, based on [1], and then present an example of the behavior of a timed automaton.

Definition 1 (Timed Automaton (TA)). *Given a finite set of actions Σ, a timed automaton is a tuple (L, ℓ_0, C, E, I), where*

- *L is a finite set of locations*
- *$\ell_0 \in L$ is the initial location*
- *C is a finite set of clocks*
- *E is a finite set of edges of the form (ℓ, g, a, r, ℓ'), where*
  - *$\ell, \ell' \in L$ are the source and destination locations*
  - *$g \in \mathcal{B}(C)$ is a constraint (guard) over the set of clocks*
  - *$a \in \Sigma$ is an action*
  - *$r \in 2^C$ is a set of clocks to reset*
- *$I : L \to \mathcal{B}(C)$ assigns invariants to locations.*

The set of clock constraints $\mathcal{B}(C)$ over the set of clocks C is defined by the following abstract syntax where $x \in C$ and $n \in \mathbb{N}$.

$$g, g' ::= g \wedge g' \mid x \leq n \mid x < n \mid x = n \mid x > n \mid x \geq n$$

Timed automata are usually given in a graphical representation with locations as nodes in a graph. The example shown in Figure 5 represents a satellite orbiting Earth. An orbit takes 90 minutes, with half of the time spent in the sun, and the other half in shadow. When the satellite is in the sun, it harvests enough energy from its solar panels to be able to rotate the satellite

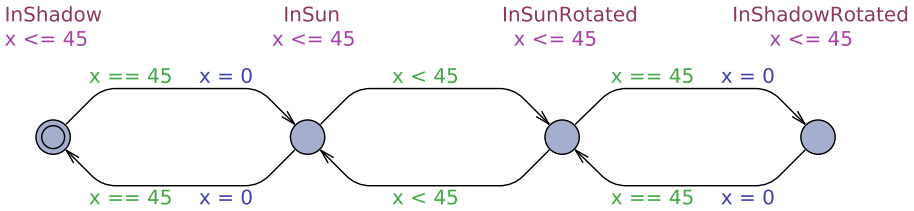

Fig. 5: A timed automaton in UPPAAL syntax representing a satellite that can rotate to aim its camera when in sunlight.

and point a camera at a point of interest. In the shadow, there is not enough energy for rotation. We will now explain the behavior of this TA informally.

This system has one clock, x, and clocks start at value 0. The double circles mark the starting location, *InShadow*. Thus, the starting state of the system is $(\mathit{InShadow}, [x \mapsto 0])$. The only edge leaving *InShadow* is guarded by $x = 45$ (which is the mathematical meaning of the computer notation `x == 45`). Therefore, the only thing that can happen is that time delays until the state $(\mathit{InShadow}, [x \mapsto 45])$ is reached. In this state, the invariant $x \leq 45$ forbids further time delays in the current location. At the same time, the guard $x = 45$ has been satisfied, so it is possible to take the edge $\mathit{InShadow} \xrightarrow{x=45,\tau,\{x\}} \mathit{InSun}$ (which is another way to write the edge $(\mathit{InShadow}, x = 45, \tau, \{x\}, \mathit{InSun})$). In taking this edge, the clock x is also reset. We will explain the action τ later.

The new state of the system is thus $(\mathit{InSun}, [x \mapsto 0])$. From this state, two options are possible: To delay time or to take the edge $\mathit{InSun} \xrightarrow{x<45,\tau,\emptyset} \mathit{InSunRotated}$. In Figure 5, no reset is specified on this edge, which means the set of clocks to reset is the empty set. While being in the sun (in the locations *InSun* or *InSunRotated*), it is possible to rotate the satellite back and forth (move between those two locations). However, when 45 minutes have elapsed, the satellite is trapped in its current state of rotation. If it is in location *InSunRotated*, it will spend the next 45 minutes in *InShadowRotated*. The same is true for the locations *InSun* and *InShadow*.

In a more realistic model, it would take some time to rotate the satellite. It would also be a great advantage to represent energy explicitly. We will see this done several times in this thesis.

As we have seen, the UPPAAL syntax uses some conventions. The set of clocks to reset is simply not specified if it is empty. Guards and invariants that are tautologies are also not specified (but can be formalized as $x \geq 0$ where x is a clock). Finally, an unspecified action becomes the "silent" action, τ.

It is called silent because no other components of a system are able to observe it. Figure 5 has but one component (automaton). But it can be easier to model a system as a collection, or *networks* of communicating automata.

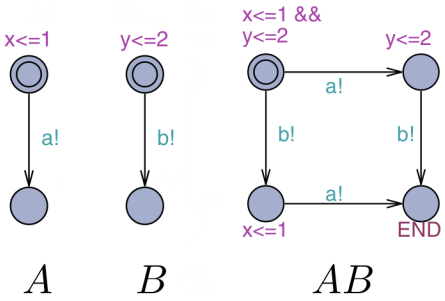

Fig. 6: Three timed automata [12].

3.1 Networks of Timed Automata

A network of timed automata is a set of automata sharing an action alphabet, Σ. Apart from τ, actions come in pairs such that two components can perform a two-way synchronization over a *channel*. For example, using the channel `ready`, one component can signal that it is ready with the action `ready!` while another can receive this signal and react to it by using the action `ready?`. If two components are in locations that have outgoing edges with the `ready!` and `ready?` actions, respectively, (and the guards are satisfied) both components can take these edges synchronously.

While a network of timed automata allows concise expression of a system, it cannot express something that a single (larger) timed automaton cannot. At least this is the case for non-deterministic semantics. But not for stochastic semantics.

3.2 Stochastic Timed Automata

Stochastic semantics can be defined for many types of timed automata, including priced and hybrid variants. Our discussion here does not depend on the precise variant. In the stochastic semantics used in UPPAAL, networks of timed automata are considered [12], which differs from the approach in [3, 6].

If two or more components are in a state where the next action is stochastic, a *race* occurs. The components each sample their delay from the relevant distribution, and the component that picks the smallest delay wins. This component then samples its output probability function. The components update their locations, and a new race starts.

To give an example, the automaton AB in Figure 6 is timed bisimilar to the composition $A|B$ in the non-deterministic setting. However, in the stochastic setting, the invariant of the initial location of AB means that the first action must be output after at most one time unit. The delay is sampled from the uniform distribution $[0, 1]$ and the output is chosen with equal weight for $a!$

and $b!$. In other words, the first output of AB will be $a!$ with probability $\frac{1}{2}$. (The stochastic setting uses broadcast synchronization such that the output $a!$ cannot be blocked waiting for other components to perform $a?$.)

In the system $A|B$, the components independently sample from their delay distributions $[0,1]$ and $[0,2]$. Component A has all its probability mass in the interval $[0,1]$ whereas B only has half its probability mass in that interval. Thus, A wins the race with probability $\frac{2}{3}$. It then outputs $a!$ with probability 1. Thus, the first output of $A|B$ will be $a!$ with probability $\frac{2}{3}$.

Networks of stochastic timed automata are used in statistical model checking (SMC). The logic property under test is monitored while the system is simulated according to the stochastic semantics for a finite number of runs. Hypothesis testing is then used to provide statistical evidence for the result (satisfaction or violation of the property) [7]. Besides hypothesis testing, SMC also supports probability estimation and probability comparison without explicit estimation (sequential testing). Individual simulations can also be studied without statistics, and histograms of values observed within a chosen number of runs can be produced.

The various features of Uppaal Smc are used in Paper A, and in Paper B, Uppaal Stratego uses the SMC simulation engine in combination with reinforcement learning to perform near-optimization.

3.3 Hybrid Automata

The final type of automata we will touch on is hybrid automata. Hybrid automata are a generalization of timed automata that uses continuous variables in place of clocks. A clock corresponds to a continuous variable evolving with a fixed rate of one. But arbitrary rates are possible in hybrid automata. Here we give a definition based on [8].

Definition 2 (Hybrid Automaton). *Given a finite set of actions Σ, a hybrid automaton is a tuple (L, ℓ_0, X, E, F, I), where*

- *L is a finite set of locations*
- *$\ell_0 \in L$ is the initial location*
- *X is a finite set of continuous variables*
- *E is a finite set of edges of the form $(\ell, g, a, \varphi, \ell')$, where*
  - *$\ell, \ell' \in L$ are locations*
  - *g is a predicate on $\mathbb{R}^X$*
  - *$a \in \Sigma$ is an action label*
  - *φ is a binary relation on $\mathbb{R}^X$*

- $F : L \to \Delta$ *assigns delay functions to locations*
- $I : L \to \mathcal{B}(X)$ *assigns invariants to locations.*

A variable valuation over X is a mapping $\nu : X \to \mathbb{R}$ (also written $\mathbb{R}^X$). Valuations evolve over time according to a delay function $\delta : \mathbb{R}_{\geq 0} \times \mathbb{R}^X \to \mathbb{R}^X$, such that $\delta(d, \nu)$ is the valuation after time d. Delay functions are assumed to be time additive, i.e. $\delta(d_2, \delta(d_1, \nu)) = \delta(d_1 + d_2, \nu)$ for all delays $d_1, d_2 \in \mathbb{R}_{\geq 0}$ and delay functions $\delta \in \Delta$. The effect of a delay function specifies a set of ordinary differential equations (ODEs) to be solved. Syntactically, a delay function is specified as the rates of change of the continuous variables. The rate expression $x' = e$ indicates that variable x changes with rate e, where the expression e may refer to constants as well as variables in X.

In Uppaal syntax, rate expressions are parts of invariants, such that an invariant could be `x <= 5 && y' == x - 2`. If no rate is specified for a variable, it is assumed to be 1.

As for the other types of automata, the semantics of a hybrid automaton is a timed labeled transition system. The states are the pairs $(\ell, \nu) \in L \times \mathbb{R}^{|X|}$ where $\nu \models I(\ell)$. There are two types of transitions. One type is the delay transition $(\ell, \nu) \xrightarrow{d} (\ell, \nu')$ where $d \in \mathbb{R}_{\geq 0}$ and $\nu' = F(\ell)(d, \nu)$. The other type is the discrete transition $(\ell, \nu) \xrightarrow{a} (\ell', \nu')$, which exists whenever there is an edge $(\ell, g, a, \varphi, \ell')$ such that $\nu \models g$ and $(\nu, \nu') \in \varphi$.

As before, this can be given both non-deterministic and stochastic interpretations. In the non-deterministic interpretation, many interesting problems are undecidable. However, in the stochastic interpretation, also known as stochastic hybrid automata, undecidable problems can be estimated using statistical model checking. Hybrid automata are used to model the Kinetic battery model as described in Section 1.2, and in Paper A.

4 Conclusion

In this thesis, we have demonstrated the applicability of statistical model checking to battery-powered systems. We asked how tools and techniques for analyzing embedded systems can be improved to analyze satellites. The main challenge for satellites is the limited energy supply combined with the behavior of batteries. Two aspects of battery behavior are important.

Getting energy out of (and into) batteries takes time, which can be described using differential equations. We showed how the *kinetic battery model* [23] lends itself to modeling as a hybrid automaton, which is naturally combined with stochastic automata to form a network of stochastic hybrid automata. This enables natural expression of complicated systems, which can be analyzed with using the technique of statistical model checking. For systems of a limited size, SMC is complementary to hybrid systems verification.

For large models, which have enormous statespaces due to the statespace explosion problem, SMC can be the only practical analysis technique.

In addition to analyzing the performance of an existing system model, SMC can be combined with reinforcement learning to synthesize control strategies that are optimized for a given criterion. Thus, instead of designing and analyzing several variants of a control program or scheduler, a near-optimal solution can be synthesized, provided the controllable parts of the system are specified. While this can help using a battery in the most efficient way in the short term, this also wears out the battery quickly. And because replacing batteries in space is prohibitively expensive, predicting battery wear can be an important aspect of satellite design and mission planning. We created the *wear score function* to rate proposed system designs on their relative impact on the long term battery life. Battery datasheets typically include endurance data for the simplest of workloads, but the wear score function takes any usage profile as input, and evaluates it on both charge/discharge rates and depth of discharge. Like the kinetic battery model, the wear score function can be integrated with a model based on timed automata.

Besides battery performance we have looked at synchronous dataflow models of the circuits that use power in a satellite. For software defined radio (SDR) circuits, we have modeled different implementations of the finite impulse response (FIR) filter as dataflow graphs. By translating these graphs to linearly priced timed automata, we have enabled the use of cost-optimal reachability algorithms, such that energy-optimal scheduling can be performed, based on power measurements of the underlying platform. We concluded that no filter implementation is the best in any case. The filters fill different parts of the time and energy trade-off.

However, while the above mentioned translation was specific for the type of circuits used in the filters, it is also useful to study general classes of systems. For the more expressive finite-state machine-based scenario-aware dataflow formalism (FSM-SADF, of which synchronous dataflow is a specific case), we have developed a general translation to timed automata. First, it enabled model checking of FSM-SADF graphs, which in turn enabled analysis of properties that existing tools cannot handle. Secondly, it enabled the combination of FSM-SADF graphs with the battery models studied previously.

Finally, we have treated a specific challenge for satellites: Computation errors. The lack of protection from the atmosphere means that computer processors in space have a considerably higher risk of computing incorrectly. We have given a method of modifying programs to guarantee detection of computation errors caused by "bitflips" (transient errors) in data registers. The method is based on a static program analysis, and a proof based on model checking. However, bitflips in the program counter register or in the encoding of the executing instruction are too unpredictable to be able to guarantee error detection. We have studied these types of errors using statistical

model checking. The results indicated that our method can reduce the risk of harmful bitflips going undetected by several orders of magnitude.

Thus, we have expanded the "toolbox" of formal methods with three tools: A method for evaluating the impact of usage profiles and task schedules on battery longevity, a method for enabling extended analysis (including in combination with battery models) of synchronous dataflow graphs, and a method for improving the resilience of control programs against bit errors caused by the conditions in space.

4.1 Future Work

Directions for further work follow the three topics summarized above. For statistical model checking involving hybrid battery models such as the Kinetic Battery model, performance can be improved using approximation techniques. Preliminary results have already been obtained in collaboration with my colleagues Peter Gjøl Jensen and Kasper Søe Luckow.

For application of battery models, another interesting venue is hybrid electric vehicles and other applications that combine several power sources with different characteristics. This can lead to interesting scheduling problems in the interplay between different power sources.

In general, it could be interesting to study a "energy storage hierarchy", similar to the memory hierarchy in computer architecture, where speed, capacity, and price are parameters. For energy storage, parameters could here also include transfer speed, capacity, and price, as well as other factors such as effifiency/waste and pollution. This idea arose in discussion with Boudewijn R. Haverkort.

The battery wear score function can rank usage profiles, but the score itself may not be proportional to the actual expected battery life. Future work includes adapting the function to predict battery wear for specific battery technologies based on measurements. Preliminary studies are being started at University of Twente by Marijn Jongerden and Boudewijn R. Haverkort. In general, the function in itself should also be validated in collaboration with researchers in energy technology or chemistry.

For dataflow formalisms, we have discussed the possibility of inventing a new scenario-aware dataflow formalism where the state machine is not driven by the dataflow graph itself, but instead by a timed automaton that independently chooses when new scenarios arrive. The consequences of such a change would be interesting to study.

Finally, we have worked on detection of computation errors due to bitflips. An interesting extension is to expand this toward the greater goal of fault recovery.

References

[1] L. Aceto, A. Ingólfsdóttir, K. G. Larsen, and J. Srba, *Reactive systems: modelling, specification and verification*. Cambridge University Press, 2007.

[2] R. Alur and D. L. Dill, "A theory of timed automata," *Theor. Comput. Sci.*, vol. 126, no. 2, pp. 183–235, Apr. 1994. [Online]. Available: http://dx.doi.org/10.1016/0304-3975(94)90010-8

[3] C. Baier, N. Bertrand, P. Bouyer, T. Brihaye, and M. Größer, "Probabilistic and topological semantics for timed automata," in *FSTTCS 2007: Foundations of Software Technology and Theoretical Computer Science, 27th International Conference, New Delhi, India, December 12-14, 2007, Proceedings*, ser. Lecture Notes in Computer Science, V. Arvind and S. Prasad, Eds., vol. 4855. Springer, 2007, pp. 179–191. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-77050-3_15

[4] C. Baier and J. Katoen, *Principles of model checking*. MIT Press, 2008.

[5] G. Behrmann, A. Cougnard, A. David, E. Fleury, K. G. Larsen, and D. Lime, "Uppaal-tiga: Time for playing games!" in *Proceedings of the 19th International Conference on Computer Aided Verification*, ser. LNCS, no. 4590. Springer, 2007, pp. 121–125.

[6] N. Bertrand, P. Bouyer, T. Brihaye, and N. Markey, "Quantitative model-checking of one-clock timed automata under probabilistic semantics," in *Fifth International Conference on the Quantitative Evaluaiton of Systems (QEST 2008), 14-17 September 2008, Saint-Malo, France*. IEEE Computer Society, 2008, pp. 55–64. [Online]. Available: http://dx.doi.org/10.1109/QEST.2008.19

[7] P. E. Bulychev, A. David, K. G. Larsen, M. Mikucionis, D. B. Poulsen, A. Legay, and Z. Wang, "UPPAAL-SMC: statistical model checking for priced timed automata," in *Proceedings 10th Workshop on Quantitative Aspects of Programming Languages and Systems, QAPL 2012, Tallinn, Estonia, 31 March and 1 April 2012.*, ser. EPTCS, H. Wiklicky and M. Massink, Eds., vol. 85, 2012, pp. 1–16. [Online]. Available: http://dx.doi.org/10.4204/EPTCS.85.1

[8] A. David, D. Du, K. G. Larsen, A. Legay, M. Mikucionis, D. B. Poulsen, and S. Sedwards, "Statistical model checking for stochastic hybrid systems," in *HSB*, ser. EPTCS, E. Bartocci and L. Bortolussi, Eds., vol. 92, 2012, pp. 122–136.

[9] A. David, P. G. Jensen, K. G. Larsen, A. Legay, D. Lime, M. G. Sørensen, and J. H. Taankvist, "On time with minimal expected cost!" in *Automated Technology for Verification and Analysis*, ser. LNCS, vol. 8837. Springer, 2014, pp. 129–145. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11936-6_10

[10] A. David, P. G. Jensen, K. G. Larsen, M. Mikucionis, and J. H. Taankvist, "Uppaal Stratego," in *Tools and Algorithms for the Construction and Analysis of Systems*, ser. LNCS, vol. 9035. Springer, 2015, pp. 206–211. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-46681-0_16

[11] A. David, K. G. Larsen, A. Legay, M. Mikucionis, and Z. Wang, "Time for statistical model checking of real-time systems," in *Proc. of Computer Aided Verification (CAV)*, ser. Lecture Notes in Computer Science, vol. 6806. Springer Verlag, 2011, pp. 349–355.

[12] A. David, K. G. Larsen, A. Legay, M. Mikučionis, D. B. Poulsen, J. V. Vliet, and Z. Wang, "Statistical model checking for networks of priced timed automata," in *FORMATS*, ser. LNCS. Springer, 2011, pp. 80–96.

[13] P. De Vidts and R. E. White, "Mathematical modeling of a nickel-cadmium cell: proton diffusion in the nickel electrode," *Journal of the Electrochemical Society*, p. 1509, 1995.

[14] M. Doyle, T. F. Fuller, and J. Newman, "Modeling of galvanostatic charge and discharge of the lithium/polymer/insertion cell," *Journal of the Electrochemical Society*, vol. 140, no. 6, pp. 1526–1533, 1993.

[15] T. F. Fuller, M. Doyle, and J. Newman, "Relaxation phenomena in lithium-ion-insertion cells," *Journal of the Electrochemical Society*, vol. 141, no. 4, pp. 982–990, 1994.

[16] ——, "Simulation and optimization of the dual lithium ion insertion cell," *Journal of the Electrochemical Society*, vol. 141, no. 1, pp. 1–10, 1994.

[17] S. Gold, "A PSPICE macromodel for lithium-ion batteries," in *Battery Conference on Applications and Advances, Twelfth Annual*. IEEE, 1997, pp. 215–222.

[18] S. C. Hageman, "Simple PSPICE models let you simulate common battery types," *EDN*, vol. 38, no. 22, p. 117, 1993.

[19] M. R. Jongerden and B. R. Haverkort, "Which battery model to use?" *IET Software*, vol. 3, no. 6, pp. 445–457, 2009.

[20] M. R. Jongerden, "Model-based energy analysis of battery powered systems," Ph.D. dissertation, University of Twente, 2010.

[21] J. Kempf, M. Bozga, and O. Maler, "As soon as probable: Optimal scheduling under stochastic uncertainty," in *Tools and Algorithms for the Construction and Analysis of Systems - 19th International Conference, TACAS 2013, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2013, Rome, Italy, March 16-24, 2013. Proceedings*, ser. Lecture Notes in Computer Science, N. Piterman and S. A. Smolka, Eds., vol. 7795. Springer, 2013, pp. 385–400. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-36742-7_27

[22] A. Legay, B. Delahaye, and S. Bensalem, "Statistical model checking: An overview," in *RV*, ser. Lecture Notes in Computer Science, vol. 6418. Springer, 2010, pp. 122–135.

[23] J. F. Manwell and J. G. McGowan, "Lead acid battery storage model for hybrid energy systems," *Solar Energy*, vol. 50, no. 5, pp. 399 – 405, 1993.

[24] E. Podlaha and H. Cheh, "Modeling of cylindrical alkaline cells vi. variable discharge conditions," *Journal of the Electrochemical Society*, vol. 141, no. 1, pp. 28–35, 1994.

[25] D. N. Rakhmatov and S. B. Vrudhula, "An analytical high-level battery model for use in energy management of portable electronic systems," in *Proceedings of the 2001 IEEE/ACM international conference on Computer-aided design*. IEEE Press, 2001, pp. 488–493.

Part II

Papers

Paper A

Battery-Aware Scheduling of Mixed Criticality Systems

Erik Ramsgaard Wognsen, René Rydhof Hansen,
Kim Guldstrand Larsen

The paper has been published in the proceedings of the
6th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA 2014) Specialized Techniques and Applications, Part II., LNCS vol. 8803, pp. 208–222, 2014.

Paper B

A Score Function for Optimizing the Cycle-Life of Battery-Powered Embedded Systems

Erik Ramsgaard Wognsen, Boudewijn R. Haverkort, Marijn Jongerden, René Rydhof Hansen, Kim Guldstrand Larsen

The paper has been published in the proceedings of the
13th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS 2015), LNCS vol. 9268, pp. 305–320, 2015.

Paper C

Energy-Aware Scheduling of FIR Filter Structures

Erik Ramsgaard Wognsen, René Rydhof Hansen,
Kim Guldstrand Larsen, Peter Koch

The paper is to be submitted.

Paper D

Model Checking of Finite-state Machine-based Scenario-aware Dataflow Using Timed Automata

Mladen Skelin, Erik Ramsgaard Wognsen, Mads Chr. Olesen,
René Rydhof Hansen, Kim Guldstrand Larsen

The paper has been published in the proceedings of the
10th IEEE International Symposium on Industrial Embedded Systems (SIES 2015), pp. 235–244, 2015.

Paper E

Formal Methods for Modelling and Analysis of Single-Event Upsets

René Rydhof Hansen, Kim Guldstrand Larsen,
Mads Chr. Olesen, Erik Ramsgaard Wognsen

The paper will appear as a workshop paper in the proceedings of the
16th IEEE International Conference on Information Reuse and Integration (IRI 2015)

ISSN (online): 2246-1248
ISBN (online): 978-87-7112-451-4

AALBORG UNIVERSITY PRESS